

A Factorization Algorithm for G -Algebras and Applications

Albert Heinle
David R. Cheriton School of Computer Science
University of Waterloo
200 University Avenue West
Waterloo, ON N2L 3G1, Canada
ahleinle@uwaterloo.ca

Viktor Levandovskyy
Lehrstuhl D für Mathematik
RWTH Aachen University
Pontdriesch 16
52062 Aachen, Germany
levandov@math.rwth-aachen.de

ABSTRACT

It has been recently discovered by Bell, Heinle and Levandovskyy that a large class of algebras, including the ubiquitous G -algebras, are finite factorization domains (FFD for short).

Utilizing this result, we contribute an algorithm to find all distinct factorizations of a given element $f \in \mathcal{G}$, where \mathcal{G} is any G -algebra, with minor assumptions on the underlying field.

Moreover, the property of being an FFD, in combination with the factorization algorithm, enables us to propose an analogous description of the factorized Gröbner basis algorithm for G -algebras. This algorithm is useful for various applications, e.g. in analysis of solution spaces of systems of linear partial functional equations with polynomial coefficients, coming from \mathcal{G} . Additionally, it is possible to include inequality constraints for ideals in the input.

Keywords

G -Algebras, Gröbner Bases, Factorization

1. INTRODUCTION

Notations: Throughout the paper we denote by \mathbb{K} a field. In the algorithmic part we will assume \mathbb{K} to be a computable field. $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ is the set of natural numbers including zero. For a \mathbb{K} -algebra R we denote by $U(R)$ the group of invertible (unit) elements of R , which is nonabelian in general. For $f \in R$ we denote by Rf the left ideal, generated by f . The main focus in this paper lies in so called G -algebras, which are defined as follows.

DEFINITION 1. For $n \in \mathbb{N}$ and $1 \leq i < j \leq n$ consider the units $c_{ij} \in \mathbb{K}^*$ and polynomials $d_{ij} \in \mathbb{K}[x_1, \dots, x_n]$. Suppose, that there exists a monomial total well-ordering \prec on $\mathbb{K}[x_1, \dots, x_n]$, such that for any $1 \leq i < j \leq n$ either $d_{ij} = 0$ or the leading monomial of d_{ij} is smaller than $x_i x_j$ with respect to \prec . The \mathbb{K} -algebra $A := \mathbb{K}\langle x_1, \dots, x_n \mid \{x_j x_i =$

$c_{ij} x_i x_j + d_{ij} : 1 \leq i < j \leq n\} \rangle$ is called a G -algebra, if $\{x_1^{\alpha_1} \cdots x_n^{\alpha_n} : \alpha_i \in \mathbb{N}_0\}$ is a \mathbb{K} -basis of A .

G -algebras [1, 19] are also known as algebras of solvable type [18, 20] and as PBW algebras [4, 5]. G -algebras are Noetherian domains of finite global, Krull and Gel'fand-Kirillov dimensions.

We assume that the reader is familiar with the basic terminology in the area of Gröbner bases, both in the commutative as well as in the non-commutative case. We recommend [3, 5, 19] as literature on this topic.

Recall, that $r \in R \setminus \{0\}$ is called **irreducible**, if in any factorization $r = ab$ either $a \in U(R)$ or $b \in U(R)$ holds. Otherwise, we call r reducible.

DEFINITION 2 (CF. [2]). Let A be a (not necessarily commutative) domain. We say that A is a finite factorization domain (FFD, for short), if every nonzero, non-unit element of A has at least one factorization into irreducible elements and there are at most finitely many distinct factorizations into irreducible elements up to multiplication of the irreducible factors by central units in A .

2. MOTIVATION AND APPLICATIONS

PROBLEM 1. Let A be a finite factorization domain and a \mathbb{K} -algebra. Given $f \in A \setminus (U(A) \cup \{0\})$, compute all its factorizations of the form $f = c \cdot f_1 \cdots f_n$, where $c \in U(A)$ and $f_i \in A \setminus U(A)$ are irreducible.

This paper is devoted in part to the algorithmic solution of Problem 1 for a broad class of G -algebras. With this algorithm one can approach a number of important problems, which we discuss in details.

Let A be a \mathbb{K} -algebra and $0 \neq L \subset A$ a finitely generated left ideal.

PROBLEM 2. Compute a proper left ideal $N \supsetneq L$.

Unfortunately, it is not known in general, whether the problem of left maximality of a given ideal with respect to inclusion is decidable. Therefore we are interested in the local negative form of it. Namely, if Problem 2 can be solved, then L is not left maximal. Moreover, for any N as above, we have a surjection from A/L to its proper factor-module A/N , in other words the exact sequence of left A -modules

$$0 \rightarrow N/L \rightarrow A/L \rightarrow A/N \rightarrow 0$$

which contributes to the knowledge of the structure of A/L .

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC '16 Waterloo, Ontario, Canada

Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...\$15.00.

Suppose that $f \in A \setminus \{0\}$, $f \notin L$ has finitely many factorizations $f = g_i h_i$ up to multiplication by central units, where $i \in I$, $|I| < \infty$ and $g_i, h_i \notin U(A)$. We do not require that g_i or h_i are irreducible. Suppose, that $L \subsetneq L + Af \subsetneq A$. Then $L + Af \subseteq \bigcap_{i \in I} (L + Ah_i)$, hence there is a natural surjective homomorphism of left A -modules

$$\frac{A}{L + Af} \rightarrow \frac{A}{\bigcap_{i \in I} (L + Ah_i)} \rightarrow 0$$

Relations with solution spaces: Let \mathcal{F} be an arbitrary (in particular, not necessarily finitely generated) left A -module, which one can think about as of the *space of solutions* for A -modules. Then, for fixed \mathbb{K} and an A -module M one denotes $\text{Sol}(M, \mathcal{F}) := \text{Hom}_A(M, \mathcal{F})$, which is a \mathbb{K} -vector space and an $\text{End}_A(M)$ -module [22].

By invoking the Noether-Malgrange isomorphism [22], we obtain for an the natural injective map of \mathbb{K} -vector spaces

$$0 \rightarrow \text{Sol}\left(\frac{A}{\bigcap_{i \in I} (L + Ah_i)}, \mathcal{F}\right) \rightarrow \text{Sol}\left(\frac{A}{L + Af}, \mathcal{F}\right).$$

The latter sheds light on the structure of the space of solutions of $A/(L + Af)$.

Note, that the following version of the left Chinese remainder theorem for modules holds:

THEOREM 1. *Let A be a \mathbb{K} -algebra, I a finite set of indices and $\{L_i : i \in I\}$ are left ideals in A . Consider the homomorphism*

$$A / \bigcap_{i \in I} L_i \xrightarrow{\phi} \bigoplus_{i \in I} A / L_i, \quad a + \bigcap_{i \in I} L_i \mapsto (a + L_1, \dots, a + L_{|I|}).$$

Then the following holds

- 1) ϕ is injective
- 2) if $\forall i, j \in I, i \neq j$ holds $L_i + L_j = A$, then ϕ is surjective.

Of course, one can assume that L_i are proper nonzero ideals.

In the second item of the Theorem 1 one says that the collection $\{L_i : i \in I\}$ is **left comaximal**. Then ϕ is an isomorphism and one has a finite direct sum decomposition of the module $A / \bigcap_{i \in I} L_i$. Hence, there is a direct sum decomposition of the solution spaces

$$\text{Sol}\left(A / \bigcap_{i \in I} L_i, \mathcal{F}\right) = \text{Sol}\left(\bigoplus_{i \in I} A / L_i, \mathcal{F}\right) = \bigoplus_{i \in I} \text{Sol}(A / L_i, \mathcal{F}).$$

Note, that the right hand side can be a direct sum even if the condition (2) is not satisfied, see Example 3.

Another application: Let A be a domain and $S \subset A$ be multiplicatively closed Ore set. By Ore's Theorem the localization $S^{-1}A$ exists and there is an injective homomorphism $A \rightarrow S^{-1}A$ [5].

A left ideal $L \subset A$ is called **left S -closed** if $L^S = L$, where $L^S := \{a \in A \mid \exists s \in S \text{ } sa \in L\} \supseteq L$ is the **S -closure** of L . There is another characterization of S -closedness: $L^S = \ker(A \rightarrow S^{-1}(A/L))$, where the latter homomorphism of A -modules is $a \mapsto 1^{-1}a + S^{-1}L$. Then L^S/L is the S -torsion submodule of A/L and A/L^S has no S -torsion.

PROBLEM 3. *Given $S \subset A$ an Ore set and $L \subset A$, give an algorithm to compute L^S .*

For a general S , it is unknown, whether L^S is computable. If A is the n th Weyl algebra, $S = \mathbb{K}[x_1, \dots, x_n] \setminus \{0\}$ and $L \subset A$ has finite holonomic rank, then there is an algorithm [24, 23] to compute L^S (known as the **Weyl closure** of L).

The factorization can be used in the process of computing L^S as follows. Let A be an FFD. Given $\ell \in L$, one computes finitely many factorizations $\ell = a_i b_i, i \in I$ for some finite indexing set I . Then let $J := \{j \in I \mid (a_j, b_j) \in S \times L\}$. If $J \neq \emptyset$, one has $L + \{Ab_j : j \in J\} \subseteq L^S$. In such a way one obtains an approximation to L^S . Note, that $L^S = A$ if and only if $L \cap S \neq \emptyset$.

3. HOW TO FACTOR IN G -ALGEBRAS

3.1 General Algorithm

In a recent publication [2, Theorem 1.3], it was proven that each G -algebra \mathcal{G} is a finite factorization domain.

In the same paper, an outline was given how one could find all possible factorizations of an element in \mathcal{G} . In this section, we will provide a thorough description of an algorithm to find all possible factorizations of an element in a G -algebra \mathcal{G} , up to multiplication by central units.

For this, we need to make a further assumption on our field \mathbb{K} , which holds for most practical choices of \mathbb{K} .

Assumption: There exists an algorithm to determine if a polynomial p in $\mathbb{K}[x]$ has roots in \mathbb{K} . If p has roots in \mathbb{K} , then this algorithm can produce all \mathbb{K} -roots of p .

Recall, that $\{x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n} : \alpha \in \mathbb{N}_0^n\}$ is a \mathbb{K} -basis of \mathcal{G} . With respect to an admissible monomial ordering \prec on \mathcal{G} we can uniquely write every $g \in \mathcal{G} \setminus \{0\}$ as $g = c_\alpha x^\alpha + t_g$ with $c_\alpha \in \mathbb{K} \setminus \{0\}$. Moreover, either $t_g = 0$ or $x^\beta \prec x^\alpha$ for any summand $c_\beta x^\beta$, $c_\beta \neq 0$ of t_g . Then $\text{lm}(g) = x^\alpha$ is the *leading monomial* of g and $\text{lc}(g) = c_\alpha$ is the *leading coefficient* of g . A polynomial $g \neq 0$ with $\text{lc}(g) = 1$ is called *monic*.

It is important to recall [19], that $\text{lm}(x^\alpha \cdot x^\beta) = x^{\alpha+\beta}$ holds $\forall \alpha, \beta \in \mathbb{N}_0^n$ in a G -algebra.

PROOF OF ALGORITHM 1. Let us begin with discussing the termination. The set M in line 2 is finite, as it is a permutation of a finite product of the variables in \mathcal{G} . Since \mathcal{G} is a G -algebra, the set of total well-orderings on it, satisfying the Definition 1, is nonempty. By [4], in this set there is a weighted degree total ordering, say \prec_w with strictly positive weights. Without loss of generality let us assume this is the ordering we are working with. Thus for any monomial there are only finitely many monomials which are smaller with respect to \prec_w . In particular, this applies to $\text{lt}(a)$ and $\text{lt}(b)$ in line 5. The variety V will be a finite set due to the fact that \mathcal{G} is an FFD. Thus, the set R in line 9 will also be finite. The recursive call will also terminate, since in each step we either discover that we cannot refine our factorization any more, or we split a given factor into two factors of strictly smaller degrees.

For the correctness discussion of our algorithm, we need to show that we can calculate the variety V in line 8. We know, since \mathcal{G} is an FFD, that the ideal generated by F is either zero-dimensional over $\mathbb{K}[x]$ or it is an intersection of such with a higher-dimensional ideal H , whereas the variety of H does not contain points from an affine space over \mathbb{K} . Hence we proceed with the zero-dimensional component F_0 of F .

Algorithm 1 Factoring an element g in a G -algebra \mathcal{G}

Input: $g \in \mathcal{G} \setminus \mathbb{K}$.

Output: $\{(g_1, \dots, g_m) \mid m \in \mathbb{N}, g_i \in \mathcal{G} \setminus \mathbb{K} \text{ for } i \in \{1, \dots, m\}, g_1 \cdots g_m = g\}$ (up to multiplication of each factor by a central unit).

Assumption: An admissible monomial ordering \prec on \mathcal{G} is fixed and g is monic with respect to it.

```
1:  $R := \{\}$ 
2:
    $M := \{(p_1, \dots, p_\nu) \mid \nu \in \mathbb{N}, p_i \in \{x_1, \dots, x_n\},$ 
        $\text{lm}(p_1 \cdots p_\nu) = \text{lm}(g)\}$ 

3: for  $(p_1, \dots, p_\nu) \in M$  do
4:   for  $i := 1$  to  $\nu - 1$  do
5:     Set up an ansatz for the  $\mathbb{K}$ -coefficients of  $a \cdot b = g$ 
       with  $\text{lt}(a) = p_1 \cdots p_i$  and  $\text{lt}(b) = p_{i+1} \cdots p_\nu$ .
6:      $F :=$  the reduced Gröbner basis w.r.t. an elimination
       ordering of the ideal generated by the coefficients
       of  $a \cdot b - g$ .
7:     if  $F \neq \{1\}$  then
8:        $V :=$  Variety of  $\langle F \rangle$  in an affine space over  $\mathbb{K}$ .
9:        $R := R \cup \{(a, b) \mid a, b \in \mathcal{G}, a \cdot b = g, \text{ where the}$ 
           coefficients of  $a, b$  are given by  $v \in V\}$ 
10:    end if
11:  end for
12: end for
13: if  $R = \{\}$  then
14:   return  $\{(g)\}$ 
15: else
16:   Recursively factor  $a$  and  $b$  for each  $(a, b) \in R$ .
17: end if
18: return  $R$ 
```

Our assumption above states that we can find all \mathbb{K} -roots of a univariate polynomial. Since F_0 is zero-dimensional, for any variable x_i there is the corresponding univariate polynomial, generating the principal ideal $F_0 \cap \mathbb{K}[x_i]$. By backwards substitution, we obtain the entire \mathbb{K} -variety of the ideal generated by F_0 . \square

EXAMPLE 1. Let us consider the universal enveloping algebra $U(\mathfrak{sl}_2)$ of \mathfrak{sl}_2 [11], represented by

$$\mathbb{K}\langle e, f, h \mid fe = ef - h, he = eh + 2e, hf = fh - 2f \rangle.$$

In $U(\mathfrak{sl}_2)$, we want to factorize the element

$$p := e^3 f + e^2 f^2 - e^3 + e^2 f + 2ef^2 - 3e^2 h - 2efh - 8e^2 \\ + ef + f^2 - 4eh - 2fh - 7e + f - h.$$

We fix the lexicographic ordering on $U(\mathfrak{sl}_2)$, i.e. the leading term of p is $e^3 f$.

Therefore the set M in line 2 is given as

$$M := \{(e, e, e, f), (e, e, f, e), (e, f, e, e), (f, e, e, e)\}.$$

When choosing (e, e, e, f) , for $i = 1$ one obtains the factorization

$$p = (e + 1) \cdot (e^2 f + ef^2 - 3eh - 2fh - e^2 + f^2 - 7e + f - h).$$

By picking (e, e, f, e) , for $i = 3$ one obtains two more fac-

torizations, namely

$$p = (e^2 f + 2ef - 2eh - e^2 - 4e + f - 2h - 3) \cdot (e + f)$$

and

$$p = (e^2 f + ef^2 - 2eh - e^2 + f^2 - 3e - f - 2h) \cdot (e + 1).$$

All the other combinations either produce the same factorizations or none.

When recursively calling the algorithm for each factor in the found factorizations, we discover that the first two factorizations have a reducible factor. In the end, one obtains the following two distinct factorizations of p into irreducible factors:

$$p = (e^2 f + ef^2 - 2eh - e^2 + f^2 - 3e - f - 2h) \cdot (e + 1) \\ = (e + 1) \cdot (ef - e + f - 2h - 3) \cdot (e + f).$$

3.2 Implementation

We have developed an experimental implementation of Algorithm 1 in the computer algebra system SINGULAR [10]. We will make it available as part of `ncfactor.lib`. Our newly implemented procedures factorize elements in any G -algebra, whose ground field is $\mathbb{F}(q_1, \dots, q_n)$, where \mathbb{F} is either \mathbb{Q} or a finite prime field and q_i are transcendental over \mathbb{F} .

We designed the software in a modular way, so that during runtime our function checks if a more efficient factorization algorithm is available for the specific given G -algebra and/or input polynomial. If this is the case, the input is re-directed to this function. In this way, the user can call the general function to factor elements in any one of the supported G -algebras, and runs the available optimized algorithms, where available, without calling them individually.

3.3 Possible Improvements

Algorithm 1 solves the problem of finding all possible factorizations of an element in a G -algebra, but it will not be very efficient in general. This is not only due to the complexity of the necessary calculation of a Gröbner basis [21], but also the size of the set M is a bottleneck. In [12, 13], an algorithm for factoring elements in the n th Weyl algebra is presented, which is similar to Algorithm 1. The main difference is that the \mathbb{Z}^n -graded structure is utilized. There, the homogeneous polynomials of degree zero form a \mathbb{K} -algebra $A_n^{(0)}$, which is isomorphic to a commutative multivariate polynomial ring. The set of homogeneous polynomials of degree $z \in \mathbb{Z}^n \setminus \{0^n\}$ has the structure of a cyclic $A_n^{(0)}$ -bimodule. Hence, factorization of homogeneous polynomials with respect to the \mathbb{Z}^n -grading reduces to factoring commutative polynomials with minor additional combinatorial steps. An inhomogeneous polynomial f has now the highest graded part $\alpha(f)$ and the lowest graded part $\omega(f)$, both of them rather polynomials than monomials. Hence $\alpha(f), \omega(f)$ have potentially smaller numbers of different factorizations than the permutations of the leading term collected in M in Algorithm 1. Indeed, it suffices to consider firstly factorizations into two polynomials and for each candidate pair an ansatz is made for the graded terms between the highest and the lowest graded parts. This means, that the set M has smaller size in general when using this technique. Additionally, this approach takes the lowest graded part into account, which allows to eliminate certain invalid cases beforehand. The performance increase is reflected by the benchmarks presented in [12, 17].

Hence, for practical implementations of Algorithm 1, one should examine each possible G -algebra separately and take advantage of potential extra structure, like the presence of nontrivial \mathbb{Z}^n -grading or an isomorphism to an algebra with this structure.

We will conclude this section by summarizing the conditions that can lead to an improved version of Algorithm 1. Let A be a \mathbb{K} -algebra, which possesses a nontrivial (i.e. not all weight vectors are zero) \mathbb{Z}^n -(multi)grading. Then one can infer the following additional information:

1. For $z \in \mathbb{Z}^n$, $A_z := \{a \in A : \deg(a) = z\} \cup \{0\}$ is a \mathbb{K} -vector space. Moreover, $\bigoplus_z A_z = A$ and $A_i A_j \subseteq A_{i+j}$ for all $i, j \in \mathbb{Z}^n$.
2. A_{0^n} , the graded part of degree zero, is a \mathbb{K} -algebra itself (since $A_0 A_0 \subseteq A_0$).
3. For $z \in \mathbb{Z}^n \setminus \{0^n\}$, the z -th graded part A_z is an A_{0^n} -bimodule (since $A_0 A_z, A_z A_0 \subseteq A_z$).

In order to be useful for factorizing purpose, this grading should have the following properties:

4. The graded part of degree zero, A_{0^n} , which is a \mathbb{K} -algebra, is additionally an FFD with "easy" factorization, preferably the commutative polynomial ring. Furthermore, for keeping the set M in Algorithm 1 small, it would be desirable if in A_{0^n} a randomly chosen polynomial is irreducible with high probability.
5. The irreducible elements in A_{0^n} , that are reducible in A , can be identified and factorized in an efficient manner. Preferably, one has a finite number of monic elements of such type.
6. For $z \in \mathbb{Z}^n \setminus \{0^n\}$, the z -th graded part A_z is a finitely generated A_{0^n} -bimodule, preferably a cyclic bimodule.

Then the Algorithm 1 can be modified along the lines of algorithms from [12, 13], which we have also sketched above. Let us illustrate this approach by a concrete example.

EXAMPLE 2. As in Example 1, let $A = U(\mathfrak{sl}_2)$, that is

$$A = \mathbb{K}\langle e, f, h \mid fe = ef - h, he = eh + 2e, hf = fh - 2f \rangle.$$

At first, let us determine which gradings are possible. Let w_e, w_f and w_h be the weights of the variables, not all zero. The two last relations of A imply that $w_h = 0$, and the first one implies $w_e + w_f = w_h = 0$, that is $w_f = -w_e$. Hence a \mathbb{Z} -grading $(w_e, w_f, w_h) = (1, -1, 0)$ is enough for our purposes, since $A_0 = \mathbb{K}[ef, h]$ is commutative and the z -th graded part is a cyclic A_0 -bimodule, generated by e^z if $z > 0$ and by $f^{|z|}$ otherwise. This property guarantees, that $\forall r \in \mathbb{K}[ef, h]$ and $\forall z \in \mathbb{N}$ there exists $q_1, q_2 \in \mathbb{K}[ef, h]$, such that $re^z = e^z q_1$ and $e^z r = q_2 e^z$ and the same holds for the multiplication by f^z . Note, that $\deg(q_i) = \deg(r)$.

We claim that the only monic irreducible elements in A_0 , which are reducible in A , are given by ef and $ef - h$. The proof to this claim is similar to the one for [13, Lemma 2.4], which we outline here: Let p be an irreducible element in A_0 , which reduces into $p = \varphi \cdot \psi$ in A , where $\varphi, \psi \in A \setminus \mathbb{K}$ are monic. Since A is a domain, the factors φ, ψ are homogeneous with $\deg(\varphi) = k$ and $\deg(\psi) = -k$ for some $k \in \mathbb{Z}$. If $|k| > 1$ or $k = 0$, p would be reducible in A_0 , which violates our assumption. Hence only $k = 1$ is possible. If

any of φ or ψ would have a non-trivial A_0 factor, we would obtain again that p is reducible in A_0 . This leaves as only options $p = ef$ or $p = fe = ef - h$, as claimed. Thus, we have shown that irreducible elements in A_0 , which are reducible in A , can be easily identified and factored.

Now consider the same polynomial p as in Example 1. With respect to the $(1, -1, 0)$ -grading it decomposes into the following graded parts: $\alpha(p) = -e^3$, $\omega(p) = f^2$ (as we see, in this case we have monomials in graded parts, while in general rather polynomials appear) and the intermediate parts are

$$\begin{aligned} & \underbrace{e^3 f - 3e^2 h - 8e^2}_{\deg:2} + \underbrace{e^2 f - 4eh - 7e}_{\deg:1} + \\ & + \underbrace{e^2 f^2 - 2efh + ef - h}_{\deg:0} + \underbrace{2ef^2 - 2fh + f}_{\deg:-1}. \end{aligned}$$

Among the factorizations of $\alpha(p) = -e^3$ and $\omega(p) = f^2$ into two factors, consider the case $(-e^2) \cdot e$ and $f \cdot f$. Thus, we're looking for $a, b \in A$ with $\alpha(a) = e^2, \omega(a) = f$ and $\alpha(b) = e, \omega(b) = f$ and $p = ab$ holds. In b we have only one possible intermediate graded part $b_0(ef, h)$, namely of degree 0 since $\deg \alpha(b) = 1$ and $\deg \omega(b) = -1$. In a we have to specify the parts of degrees 1 resp. 0, that is $a_1(ef, h) \cdot e$ resp. $a_0(ef, h)$. After the multiplication, we obtain the following graded decomposition of intermediate graded terms of ab :

$$\begin{aligned} & \underbrace{-e^2 b_0 + a_1 e^2}_{\deg:2} + \underbrace{a_1 e b_0 + a_0 e - e^2 f}_{\deg:1} + \\ & + \underbrace{a_1 e f + a_0 b_0 + ef - h}_{\deg:0} + \underbrace{f b_0 + a_0 f}_{\deg:-1}. \end{aligned}$$

By fixing the maximal possible degree of $a_0, a_1, b_0 \in \mathbb{K}[ef, h]$, we can create and solve a system of equations which the coefficients of a_0, a_1, b_0 have to satisfy. In this example an ansatz in terms of $1, h, ef$, i.e. 9 unknown coefficients, leads to the system of 18 at most quadratic equations, which leads to the unique solution: $b_0(ef, h) = 0$, $a_0(ef, h) = 2ef - 2h - 3$ and $a_1(ef, h) = ef - h - 2$. Substituting the polynomials, we arrive at the following factorization with polynomials sorted according to the grading:

$$p = (-e^2 + e^2 f - 2eh - 4e + 2ef - 2h - 3 + f) \cdot (e + f)$$

This is already known to us from the Example 1. In an analogous way one can address other factorizations. Note, that in the ansatz we made, significantly less variables for unknown coefficients and a system of less equations of smaller total degree were used, compared to the general Algorithm.

4. THE FACTORIZED GRÖBNER BASIS ALGORITHM FOR G -ALGEBRAS

In what follows, by the term ideal we always mean a left ideal (unless otherwise specified).

The factorized Gröbner approach has been studied extensively for the commutative case [8, 7, 9, 14, 15], and implementations are e.g. provided in the computer algebra systems SINGULAR [10] and REDUCE [16].

The leading motivation is to split a Gröbner basis computation into smaller pieces with respect to the degrees of their generators. The union of the varieties of the ideals generated by these smaller pieces equals the variety of the initial system.

In the commutative case, there is also a way to constrain the solution space. One can provide an extra set of elements, that should not be reducible by the computed Gröbner bases. In this way, one excludes certain unwanted solutions, which is useful in practice.

The search for varieties in the commutative case translates to the search for solutions in the non-commutative case: All G -algebras are finite factorization domains and a general factorization algorithm via Algorithm 1 is given. Many of them are abstractions of algebras of operators, and one is interested to find common solutions of certain sets of operators, written as polynomials. Right hand factors of elements correspond to partial solutions, and hence a split similar to the commutative case is helpful to recover partial solutions. Motivated by this observation, we attempt to generalize the factorized Gröbner basis algorithm to the G -algebra case in this section. Our algorithm includes the possibility to introduce constraints, similar to the methods in the commutative case.

Unfortunately, not all nice properties transfer into the non-commutative case, as the following example depicts.

EXAMPLE 3. *In the commutative case, one has the property that the radical of the input ideal will be equal to the intersection of the radicals of all ideals computed by the factorized Gröbner basis algorithm.*

We will show via a counter-example that we do not have the same property for G -algebras.

Consider

$$p = (x^6 + 2x^4 - 3x^2)\partial^2 - (4x^5 - 4x^4 - 12x^2 - 12x)\partial + (6x^4 - 12x^3 - 6x^2 - 24x - 12) \in A_1.$$

This polynomial appears in [24] and has two different factorizations, namely

$$\begin{aligned} p &= (x^4\partial - x^3\partial - 3x^3 + 3x^2\partial + 6x^2 - 3x\partial - 3x + 12) \cdot \\ &\quad (x^2\partial + x\partial - 3x - 1) \\ &= (x^4\partial + x^3\partial - 4x^3 + 3x^2\partial - 3x^2 + 3x\partial - 6x - 3) \cdot \\ &\quad (x^2\partial - x\partial - 2x + 4) \end{aligned}$$

A reduced Gröbner basis of $\langle x^2\partial + x\partial - 3x - 1 \rangle \cap \langle x^2\partial - x\partial - 2x + 4 \rangle$, computed in SINGULAR [10], is given by

$$\begin{aligned} &\{3x^5\partial^2 + 2x^4\partial^3 - x^4\partial^2 - 12x^4\partial + x^3\partial^2 - 2x^2\partial^3 + 16x^3\partial \\ &\quad + 9x^2\partial^2 + 18x^3 + 4x^2\partial + 4x\partial^2 - 42x^2 - 4x\partial - 12x - 12, \\ &\quad 2x^4\partial^4 - 2x^4\partial^3 + 11x^4\partial^2 + 12x^3\partial^3 - 2x^2\partial^4 - 2x^3\partial^2 \\ &\quad + 10x^2\partial^3 - 44x^3\partial - 17x^2\partial^2 + 64x^2\partial + 12x\partial^2 + 66x^2 \\ &\quad + 52x\partial + 4\partial^2 - 168x - 16\partial - 60\}. \end{aligned}$$

Hence, one main difference will be that we do not claim that the union of all solutions of our smaller pieces in the factorizing Gröbner basis algorithm will always be equal to all common solutions of the initial set of polynomials. In general, we will only find a subset of all solutions using our method. However, in this example, the space of holomorphic solutions of the differential equation associated to p in fact coincides with the union of the solution spaces of the two generators of the intersection stated above.

DEFINITION 3. *Let B, C be finite subsets in \mathcal{G} . We call the tuple (B, C) a **constrained Gröbner tuple**, if B is a*

Algorithm 2 Factorized Gröbner bases Algorithm for G -Algebras (FGBG)

Input: $B := \{f_1, \dots, f_k\} \subset \mathcal{G}$, $C := \{g_1, \dots, g_l\} \subset \mathcal{G}$.

Output: $R := \{(\tilde{B}, \tilde{C}) \mid$

$(\tilde{B}, \tilde{C}) \text{ is factorized constrained Gröbner tuple} \text{ with } \langle \tilde{B} \rangle \subseteq \bigcap_{(\tilde{B}, \tilde{C}) \in R} \langle \tilde{B} \rangle$

Assumption: All elements in B and C are monic.

```

1: for  $i = 1$  to  $k$  do
2:   if  $f_i$  is reducible then
3:      $M := \{(f_i^{(1)}, f_i^{(2)}) \mid f_i^{(1)}, f_i^{(2)} \in \mathcal{G} \setminus \mathbb{K}, \text{lc}(f_i^{(1)}) =$ 
        $\text{lc}(f_i^{(2)}) = 1, f_i^{(1)} \cdot f_i^{(2)} = f_i, f_i^{(1)} \text{ is irreducible}\}$ 
4:     if there exists  $(a, b), (\tilde{a}, \tilde{b}) \in M$  with  $\tilde{a} \neq a$  then
5:       return
6:   end if
7: end if
8: end for
9:  $P := \{(f_i, f_j) \mid i, j \in \{1, \dots, k\}, i < j\}$ 
10: while  $P \neq \emptyset$  do
11:   Pick  $(f, g) \in P$ 
12:    $P := P \setminus \{(f, g)\}$ 
13:    $s :=$  S-polynomial of  $f$  and  $g$ 
14:    $h := \text{NF}(s, B)$ 
15:   if  $h \neq 0$  then
16:     if  $h$  is reducible then
17:       return FGBG( $B \cup \{h\}, C$ )
18:     end if
19:      $P := P \cup \{(h, f) \mid f \in B\}$ 
20:      $B := B \cup \{h\}$ 
21:   end if
22:   if there exists  $i \in \{1, \dots, l\}$  with  $\text{NF}(g_i, B) = 0$  then
23:     return  $\emptyset$ 
24:   end if
25: end while
26: return  $\{(B, C)\}$ 

```

*Gröbner basis of $\langle B \rangle$, and $\text{NF}(g, B) \neq 0$ for every $g \in C$. We call a constrained Gröbner tuple **factorized**, if every $f \in B$ is either irreducible or has a unique irreducible left divisor.*

It is possible to strengthen the assumptions on a factorized constrained Gröbner tuples by only allowing completely irreducible elements in B , which might be preferable depending on the concrete problem. However, in our application, we allow elements with only one factorization. In this way, we increase the number of solutions we can find for a certain system $B \subset \mathcal{G}$ by using our generalized factorized Gröbner basis algorithm. This methodology also appears in the context of semifirs, where the concept of so called block factorizations or cleavages has been introduced to study the reducibility of a principal ideal [6, Chapter 3.5].

PROOF OF ALGORITHM 2. We will first discuss the termination aspect of Algorithm 2. Since M as calculated in line 3 is of finite cardinality, the existence check in line 4 can be done in a finite number of steps. Line 5 consists of a finite

number of recursive calls to FGBG. The algorithm reaches this line if there is an element f in B , which is reducible and has a non-unique irreducible left divisor. In each recursive call, the algorithm is called with an altered version of the set B , where f is being replaced in B by $b \in \mathcal{G}$, where b is chosen such that there exists an irreducible a in \mathcal{G} with $f = ab$. Therefore, after a finite depth of recursion, FGBG will be called with a set B containing elements that are either irreducible or have an unique irreducible left divisor. We can make this assumption on B when FGBG reaches line 9. Lines 10–25 describe the Buchberger algorithm to compute a Gröbner basis, with two differences:

1. If the normal form h of an S-polynomial with respect to B is not 0, we check h for reducibility. If h is reducible, we call FGBG recursively, adding h to B .
2. We check the system for consistency, i.e. if there is an element in C that reduces with respect to B , we return the empty set.

Each recursive call will terminate, since we add an element to B that will reduce an S-polynomial to zero, which could not be reduced to zero before.

For the correctness discussion, one observes that lines 1–8 serve the purpose to split the computation based on the reducibility of the elements in the initial set B . If an element $f \in B$ factorizes in more than one way, we recursively call FGBG with $(B \setminus \{f\}) \cup \{b\}$ as the generator set for each maximal right hand factor b of f . Hence, the left ideal generated by $(B \setminus \{f\}) \cup \{b\}$ will contain $\langle B \rangle$, and thus $\langle B \rangle$ will be contained in the intersection of all of them, as required.

As already mentioned in the termination discussion, lines 10–25 describe the Buchberger algorithm. After computing an S-polynomial h , we check for its reducibility. If there is more than one maximal right factor r of h , we call FGBG recursively and add h to our set B . Here, we have again a guarantee that the left ideal generated by B is a subset of the left ideal generated by $B \cup \{h\}$.

The additional constraints that we impose on each recursive call enable us to minimize our computations, but do not violate the subset property. In the end, it is ensured that in all computed constrained Gröbner tuples (\tilde{B}, \tilde{C}) , no element in C lies in the left ideal generated by \tilde{B} . \square

EXAMPLE 4. Let us execute FGBG on an example. Let

$$B := \{\partial^4 + x\partial^2 - 2\partial^3 - 2x\partial + \partial^2 + x + 2\partial - 2, \\ x\partial^3 + x^2\partial - x\partial^2 + \partial^3 - x^2 + x\partial - 2\partial^2 - x + 1\}$$

be a subset of the first Weyl algebra A_1 . We assume that $C := \{\partial - 1\}$, and that our ordering is the degree reverse lexicographic one with $\partial > x$. This example is taken from the SINGULAR manual [10] (and it is a Gröbner basis for the left ideal $\langle \partial^2 + x \rangle \cap \langle \partial - 1 \rangle$; hence we would expect the output with our chosen C to be $\langle \partial^2 + x \rangle$). Each element factors separately as

$$f_1 := \partial^4 + x\partial^2 - 2\partial^3 - 2x\partial + \partial^2 + x + 2\partial - 2 \\ = (\partial^3 + x\partial - \partial^2 - x + 2) \cdot (\partial - 1) \\ = (\partial - 1) \cdot (\partial^3 + x\partial - \partial^2 - x + 1)$$

respectively

$$f_2 := x\partial^3 + x^2\partial - x\partial^2 + \partial^3 - x^2 + x\partial - 2\partial^2 - x + 1 \\ = (x\partial^2 + x^2 + \partial^2 + x - \partial - 1) \cdot (\partial - 1) \\ = (x\partial - x + \partial - 2) \cdot (\partial^2 + x).$$

Hence, in line 5, FGBG will return two recursive calls of itself, namely

- FGBG($\{\partial - 1, f_2\}, \{\partial - 1, \partial^3 + x\partial - \partial^2 - x + 1\}$)
- FGBG($\{\partial^3 + x\partial - \partial^2 - x + 1, f_2\}, C$)

For simplicity, we will ignore the first call, as C contains $\partial - 1$, which also appears in the generator list.

Furthermore, the new element $b_1 := \partial^3 + x\partial - \partial^2 - x + 1$ only has one possible factorization. Therefore, we consider now the factorizations of f_2 . This leads again in line 5 to two recursive calls:

- FGBG($\{b_1, \partial - 1\}, \{\partial - 1, \partial^2 + x\}$)
- FGBG($\{b_1, \partial^2 + x\}, C$)

As before, we can ignore the first recursive call. Thus, we are left with $(\{b_1, \partial^2 + x\}, C)$ to proceed on line 9.

The normal form of the S-polynomial of b_1 and $\partial^2 + x$ is equal to zero. Further, the normal form of b_1 with respect to $\langle \partial^2 + x \rangle$, is equal to zero, i.e. $\partial^2 + x$ is a right divisor of b_1 . Hence, we can omit b_1 and our complete Gröbner basis is given by $\{\partial^2 + x\}$. Since $\text{NF}(\partial - 1, \langle \partial^2 + x \rangle) \neq 0$, our algorithm returns $(\{\partial^2 + x\}, C)$ as final output.

Note, that if we would have chosen $C = \emptyset$ in the beginning, the output of our algorithm would have been

$$(\{\partial - 1\}, \{b_1\}), (\{\partial^2 + x\}, \{\partial - 1\}),$$

i.e. we recover $\langle B \rangle = \langle \partial^2 + x \rangle \cap \langle \partial - 1 \rangle$ in this case.

REMARK 1. One can also insert an early termination criterion inside Algorithm 2, namely after at least one factorized constrained Gröbner tuple has been found. This is in the commutative case motivated by the fact that in practice users are often not interested in all the elements in a variety but would be content with at least one. For example, the computer algebra system REDUCE can be instructed to stop after finding one factorized Gröbner basis (see [16]). In the non-commutative case, we can only hope for partial solutions in general, but a mechanism to stop a computation once at least one is found is also desirable.

5. CONCLUSIONS

An algorithm for factoring elements in G -algebras, where the underlying field \mathbb{K} has the property that we are able to extract all possible \mathbb{K} -roots of any polynomial in $\mathbb{K}[x]$, has been shown (Algorithm 1).

This algorithm and the FFD-property of G -algebras enable us to propose a generalization of the factorized Gröbner basis algorithm for G -algebras (Algorithm 2).

A future work would be to identify improvements to Algorithm 1 for practically interesting G -algebras. This has been studied e.g. for partial q -differential, differential and difference operators in [12, 13], where the \mathbb{Z}^n graded structure resp. a certain embedding has been utilized. In the meantime, we have implemented the unimproved version in the SINGULAR library `ncfactor.lib`, which will be made

available shortly. Our implementation identifies beforehand if an improved method is already included in `ncfactor.lib` for a specific algebra and, if this is the case, re-directs the input there. This modular design allows us to update the function once an improved algorithm is available for a certain G -algebra. The use of the function stays the same after such an update.

Another interesting future direction would be to characterize further the connection between the solution space of a polynomial system $B \subset \mathcal{G}$ and the union of the solution spaces of the output of Algorithm 2 when called with B . Especially, it would be interesting to identify properties of \mathcal{G} and B , under which both spaces coincide.

An implementation of Algorithm 2 would also be of practical interest, which the authors intend to provide in the near future.

6. ACKNOWLEDGMENTS

We thank Eugene Zima, Jason P. Bell and Mark Giesbrecht for insightful discussions we had with them. We are grateful to Wolfram Koepf for the information on internals of REDUCE.

7. REFERENCES

- [1] J. Apel. Gröbnerbasen in nichtkommutativen Algebren und ihre Anwendung. *Dissertation, Universität Leipzig*, 1988.
- [2] J. P. Bell, A. Heinle, and V. Levandovskyy. On noncommutative finite factorization domains. *To Appear in the Transactions of the American Mathematical Society*; *arXiv preprint arXiv:1410.6178*, 2014.
- [3] B. Buchberger. Introduction to Gröbner Bases. In *Gröbner Bases and Applications*, pages 3–31. Berlin: Springer, 1997.
- [4] J. Bueso, J. Gómez-Torrecillas, and F. Lobillo. Re-filtering and exactness of the Gelfand-Kirillov dimension. *Bull. Sci. Math.*, 125(8):689–715, 2001.
- [5] J. Bueso, J. Gómez-Torrecillas, and A. Verschoren. *Algorithmic methods in non-commutative algebra. Applications to quantum groups*. Dordrecht: Kluwer Academic Publishers, 2003.
- [6] P. M. Cohn. *Free ideal rings and localization in general rings*, volume 3. Cambridge University Press, 2006.
- [7] S. R. Czapor. Solving algebraic equations: combining buchberger’s algorithm with multivariate factorization. *Journal of Symbolic Computation*, 7(1):49–53, 1989.
- [8] S. R. Czapor. Solving algebraic equations via buchberger’s algorithm. In *Eurocal’87*, pages 260–269. Springer, 1989.
- [9] J. H. Davenport. Looking at a set of equations. *Technical report*, pages 87–06, 1987.
- [10] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 4-0-2 — A computer algebra system for polynomial computations. <http://www.singular.uni-kl.de>, 2015.
- [11] J. Dixmier. *Enveloping algebras*, volume 14. Newnes, 1977.
- [12] M. Giesbrecht, A. Heinle, and V. Levandovskyy. Factoring linear differential operators in n variables. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, pages 194–201. ACM, 2014.
- [13] M. Giesbrecht, A. Heinle, and V. Levandovskyy. Factoring linear partial differential operators in n variables. *Journal of Symbolic Computation*, pages –, 2015.
- [14] H.-G. Gräbe. On factorized gröbner bases. In *Computer algebra in science and engineering*, pages 77–89. World Scientific. Citeseer, 1995.
- [15] H.-G. Gräbe. *Triangular systems and factorized Gröbner bases*. Springer, 1995.
- [16] A. Hearn. Reduce user’s manual, version 3.8 (2004).
- [17] A. Heinle and V. Levandovskyy. Factorization of \mathbb{Z} -homogeneous polynomials in the first (q)-weyl algebra. *arXiv preprint arXiv:1302.5674*, 2013.
- [18] A. Kandri-Rody and V. Weispfenning. Non-commutative gröbner bases in algebras of solvable type. 9(1):1–26, 1990.
- [19] V. Levandovskyy. Non-commutative computer algebra for polynomial algebras: Gröbner bases, applications and implementation. *Doctoral Thesis, Universität Kaiserslautern*, 2005.
- [20] H. Li. *Noncommutative Gröbner bases and filtered-graded transfer*. Springer, 2002.
- [21] E. W. Mayr and A. R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Advances in mathematics*, 46(3):305–329, 1982.
- [22] W. M. Seiler. *Involution. The formal theory of differential equations and its applications in computer algebra*. Algorithms and Computation in Mathematics 24. Berlin:Springer, 2010.
- [23] H. Tsai. *Algorithms for algebraic analysis*. PhD thesis, University of California at Berkeley, 2000.
- [24] H. Tsai. Weyl closure of a linear differential operator. *Journal of Symbolic Computation*, 29:747–775, 2000.